클라우드 게이밍 시스템 개발

팀명 : **Tree**

정보컴퓨터공학부 202155581 이수빈
정보컴퓨터공학부 202055574 이다은
정보컴퓨터공학부 201914130 이지민

2024년 05월 30일

지도교수 **:** 김 원 석

목차

1. 프로젝트 개요

a. 배경 및 필요성

지난 수년간 클라우드 컴퓨팅 기술은 급격한 발전을 이루어냈다. 클라우드 컴퓨팅은 데이터 저장, 처리, 관리 등의 기능을 인터넷을 통해 제공하는 기술이다. IT 자원의 유연한 활용과 효율적인 비용 관리가 가능하다는 점에서 많은 기업과 개발자들에게 주목받고 있다. 아마존 웹 서비스(AWS), 마이크로소프트 애저(Microsoft Azure)와 같은 주요 클라우드 서비스 제공업체들이 다양한 클라우드 서비스를 제공함으로써, 고성능 컴퓨팅 자원을 필요로 하는 응용 프로그램을 클라우드 환경에서 구동하는 것이 현실화되었다. 이로 인해 사용자들은 물리적인 하드웨어에 의존하지 않고도 클라우드의 강력한 연산 능력을 활용할 수 있게 되었다.

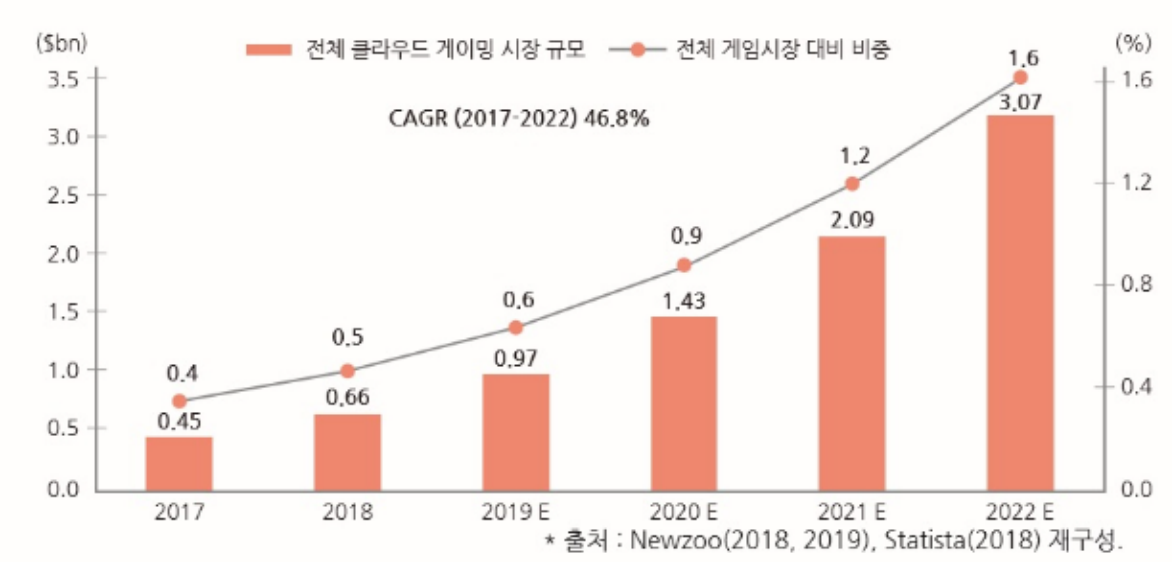

그림 1. 클라우드 게임 시장 성장률 그래프

컴퓨터 및 콘솔 하드웨어 기반의 게임은 사용자의 하드웨어 사양에 따라 서비스의 품질에 차이가 있을 수 있다. 클라우드 게이밍 기술을 활용하면 사용자들에게 개인의 하드웨어에 의존하지 않고 동일한 서비스를 제공할 수 있다. 또한 개인의 하드웨어에 의존하지 않으므로, OS마다 개발을 해야하는 크로스 플랫폼 문제를 해결할 수 있다.

b. 과제 목표 및 기대 효과

클라우드 게이밍은 클라우드 컴퓨팅 기술을 이용하여 서버에서 게임을 실행하는 기술이다. 클라우드 게이밍을 이용하면 인터넷을 통해 사용자에게 컨텐츠가 스트리밍되며, 특정 하드웨어에 종속되지 않고 대부분의 디바이스에서 게임을 구동할

수 있다. 인터넷에 의존하는 기술인 만큼, 네트워크 지연과 사용자의 트래픽 급증에 대한 유동적인 처리를 통해 서비스의 안전성을 높일 것이다. 이를 위해 저지연 스트리밍 기술을 구현하고, 사용자들의 트래픽을 모니터링하여 안정적인 인프라를 구축하여 서비스의 신뢰성을 확보할 것이다. '클라우드 게이밍 시스템 개발'은 클라우드 게이밍 서비스를 구축하여, 안정적이고 원활한 게이밍 서비스를 제공하는 것이 목표이다.

2. 요구 조건 분석과 제약 사항

a. 요구 조건 분석

1) 클라우드 게임 (스트리밍 게임) 시스템

- **Unity Render Streaming**

  Unity를 이용하여 개발한 게임을 클라우드 및 로컬 서버에서 실행 가능해야하며, 사용자는 웹 브라우저 및 다른 기기에서 해당 게임을 플레이 할 수 있어야 한다. Unity Render Streaming 기술을 통해 스트리밍 서비스를 제공하며, 운영체제 문제, 사양 문제를 해결할 수 있다.

- **WebRTC**

  아래의 표 1을 보면 Web Socket은 채팅 시스템에 더 적합하다. HLS는 Latency가 상대적으로 길어 실시간 스트리밍 게임에 적합하지 않다. Unity WebGL은 게임의 복잡성과 유저 수에 따라 추가 프로토콜을 이용해야 한다. 따라서 Unity와 사용자 브라우저 사이를 연결하기 위해 WebRTC 기술을 활용하여 진행한다.

	Web Socket	WebRTC	HLS	Unity WebGL
Purpose	실시간 양방향 통신	실시간 미디어 통신	비디오 및 오디오 스트리밍	웹 기반 3D 게임 엔진
Apply contents	텍스트, 바이너리	오디오, 비디오	오디오, 비디오	3D 그래픽, 애니메이션
Network Lantecy	매우 낮음	매우 낮음	높음	보통
Case	채팅 App, 실시간 게임	화상 회의, 온라인 게임	대규모 비디오 스트리밍	MMO 게임

표 1. Streaming Protocol Analysis

2) 서비스 구축

- **React**
  사용자에게 제공되는 인터페이스를 구성한다.사용자 정보를 저장하기 위해 상태 관리 라이브러리인 redux를 사용하여 저장한다. 클라이언트에서 페이지를 렌더링하는 방식인 CSR를 사용하여 서버의 부하를 줄이고, 초기 렌더링 이후 빠른 페이지 전환을 제공한다.
- **Spring Webflux**
  Non - Blocking 방식으로 운영하여 효율성이 좋다. Spring Boot - Spring Webflux를 통해 요청 사항을 DB에 반영하는 API 서버를 구축한다.
- **MySQL & Redis**
  사용자 정보, 친구 정보, 방 탐색, 게임 플레이 정보, 게임 머니 결제, 랭킹 등 데이터를 저장하는 DB를 구축한다. 접근 요청이 빈번하게 발생하는 데이터는 가볍고 빠른 In - Memory Database인 Redis를 이용한다.
- **Github Actions**
  CI/CD를 구축하여 개발과 배포 과정을 자동화한다.
- **Kubernetes & Nginx**
  효율적인 연산 처리를 위해 Kubernetes를 활용하여 컨테이너를 관리하고 Nignx를 통해 사용자의 트래픽 처리, 비정상적인 요청 분류 등 트래픽을 관리한다.

b. 현실적 제약 사항 및 대책

1) 버그 대처 - 실시간 로깅 및 모니터링 시스템 구축

- 게임은 버그 대처가 신속해야 한다. 이를 위해 실시간 로깅 및 모니터링 시스템을 구축하여 관리한다.
- **ELK Stack**
  로그를 집계하고 이를 분석하여 애플리케이션과 인프라 모니터링 시각화를 생성하고, 빠르게 문제를 해결할 수 있는 환경을 구축한다.
- **Prometheus**
  시스템 및 서비스의 상태와 성능을 수집하고 저장한다. 실시간으로 모니터링하고 이력을 보존할 수 있다. 다양한 데이터 소스와 통합할 수 있으며 시스템의 상태를 추적하고 문제를 식별하는 데 활용한다.
- **Grafana**

Prometheus와 같은 데이터를 시각적으로 표현하고 분석하기 위해 사용한다. 다양한 데이터 소스를 통합하여 한 곳에서 모니터링 및 분석이 가능하도록 지원하기 때문에 종합적으로 관리하기 용이하다.

3. 프로젝트 소개 및 설계

a. 개발 환경 및 기술 스택

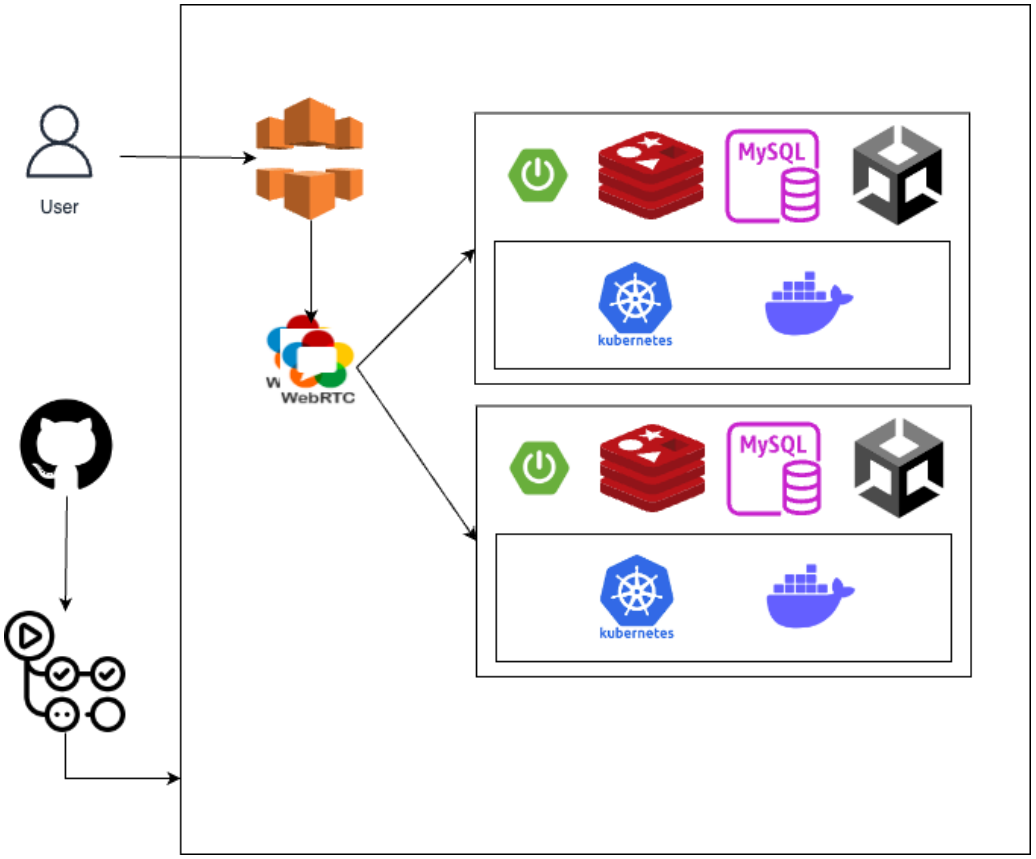

그림 2. 시스템 구성도

b. 시스템 동작 방식 설명

1) 사용자 관점

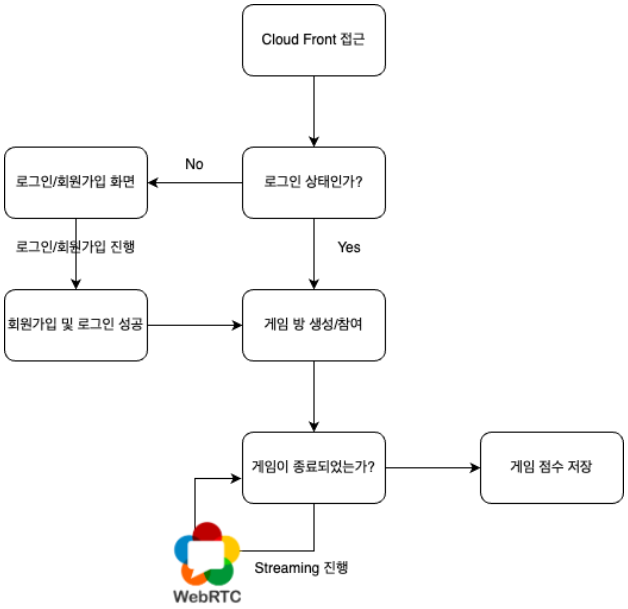

그림 3. 사용자 시나리오

- 사용자는 AWS CloudFront 에서 동작 중인 웹 서비스에 접속한다.
- 회원 가입과 로그인을 통해 게임을 실행할 수 있는 방에 접근할 수 있는 권한을 얻는다.
- 게임 방을 생성하거나 이미 생성된 게임 방에 접속한다.
- 서버에서 게임 화면을 스트리밍하고, 사용자는 웹 브라우저에서 중계 받는다.
- 게임 중 획득한 점수는 누적되고, 게임이 종료될 경우 획득한 점수가 데이터베이스에 저장된다.

2) 개발자 관점

- WebRTC Protocol, Unity Render Streaming과 Kubernetes 기술을 활용해 클라우드 환경에서 동작 가능한 게이밍 플랫폼을 제작한다.
- React 라이브러리를 이용해 게임 사용자가 스트리밍 되는 화면을 볼 수 있도록 하는 웹 어플리케이션을 제작한다.
- Spring boot와 Redis, MySQL 기술을 이용해 사용자의 게임 정보와 권한 정보가 저장될 수 있도록 API Server와 Database를 구축한다.
- Streaming Service와 여러 도메인의 서비스를 컨테이너화 시키고, 컨테이너 관리를 위해 Kubernetes를 사용한다.

- ELK Stack, Prometheus와 Grafana를 이용해 버그를 확인하고 시각화한다.
- Github actions 을 활용해 자동화 배포 파이프라인을 구축한다.

4. 개발 일정 계획 및 담당 업무

a. 개발 일정

	5월	6월	7월	8월	9월	10월
1. 일정 및 담당 정리 2. 구현 내용 구체화 3. 개발 환경 세팅 4. 아키텍처 설계 5. 착수 보고서 작성						
1. 게임 개발 part 1 2. DB 설계						
1. 게임 개발 part 2 2. Back-End 개발 3. CICD 구축						
1. Front-End 개발 2. 컨테이너 구축 3. 모니터링 구축 4. 중간 보고서 작성						
1. 로깅 시스템 구축 2. 테스트 및 디버깅 3. 최종 테스트						
1. 최종 보고서 작성 2. 결과물 업로드						

표 2. 개발 일정

b. 담당 업무

이수빈

- 컨테이너 구축 (k8s)
- 모니터링 구축 (Prometheus, Grafana)

이다은

- 게임 개발 (Unity & WebRTC)
- Front - End 개발 (React)

이지민

- Back - End 개발 (Java & Spring Boot)
- DB 구축 (MySQL & Redis)
- CICD 구축 (Github Actions)

ALL

- Unity Render Streaming 기술 바탕 주제 구체화
- Architecture 설계
- 보고서 작성

5. 참고 문헌

- 이현진, “클라우드 산업 동향 및 핵심 성장요인 분석", 한국 수출입 은행 - 해외 경쟁 연구소, 2022-02-28, https://keri.koreaexim.go.kr/HPHFOE052M01/65345?curPage=1&srchText=%ED%81%B4%EB%9D%BC%EC%9A%B0%EB%93%9C

- NVIDIA Korea, “클라우드 게이밍이란?”, NVIDIA, 2021-03-09, https://blogs.nvidia.co.kr/blog/what-is-cloud-gaming/

- 채상우, “5G시대 초고속 성장 클라우드 게임…한국은 도태!”, 헤럴드 경제, 2020-01-23, https://mbiz.heraldcorp.com/view.php?ud=20200123000024

- “About Unity Render Streaming”, Unity Manual, 2024년 5월 26일 검색, https://docs.unity3d.com/Packages/com.unity.renderstreaming@3.1/manual/index.html

- Ron Miller, “On-prem data centers are hanging in, but cloud capacity is growing much faster”, DISRUPT, 2023-07-14, https://techcrunch.com/2023/07/14/on-prem-data-centers-are-hanging-in-but-cloud-capacity-is-growing-much-faster/

- Marc Whitten, Sarah Bond, “ Unity and Microsoft announce cloud partnership for game developers and beyond”, Unity, 2022-08-08 https://blog.unity.com/news/unity-and-microsoft-announce-cloud-partnership-for-game-developers-and-beyond

- “ELK 스택이란 무엇인가요?”, AWS, 2024년 5월 26일 검색, https://aws.amazon.com/ko/what-is/elk-stack/